OperA and Brahms: a symphony?
Integrating Organizational and Emergent Views on Agent-Based Modeling

Bart-Jan van Putten[1,2], Virginia Dignum[1], Maarten Sierhuis[2], Shawn R. Wolfe[3]

[1] Utrecht University, Intelligent Systems Group,
PO Box 80089, 3508 TB Utrecht, The Netherlands
{bputten, virginia}@cs.uu.nl

[2] RIACS / NASA Ames Research Center,
Mail Stop B269-1, Moffett Field, CA 94035, USA
Maarten.Sierhuis-1@nasa.gov

[3] NASA Ames Research Center, Intelligent Systems Division,
Mail Stop B269-2, Moffett Field, CA 94035, USA
Shawn.R.Wolfe@nasa.gov

Abstract. The *organizational view* on work systems focuses on the desired outcomes of work, while the *emergent view* focuses on how the work actually gets done. Often a gap exists between these two, because workers pursue individual objectives in addition to the organizational objectives. Agent-based simulations can be used to improve work practice and thereby organizational performance. Current modeling and simulation frameworks only represent either one of the two views. In order to model both views, we propose an integration of two modeling and simulation frameworks, OperA and Brahms. Using the integrated model, we are able to run simulations that show to what degree the actual work practice differs from the organizational objectives.

1. Introduction

Organizations are intentionally formed to accomplish a set of common objectives, defined by the policy makers of the organization. People that work for those organizations often only partially pursue the global objectives of the organization. Workers often pursue their individual objectives as well, frequently resulting in a gap between the a priori designed flows of tasks and procedures reflecting the ideal activity of the organization (i.e., the *work process*), and the activities that actually get things done (i.e., the *work practice*) [1]. This gap does not exist only because of the difference in objectives between individuals and the organization, but also because many policy makers abstract from work practice when they design work systems (i.e., business operations). For example, it is uncommon for a job description to include 'socialize with co-workers', 'drink coffee', or 'read e-mail'.

Human Resource Management research has recognized that ultimately employee behaviors, rather than management practices, are the key to value creation in organizations [2]. Policy makers of successful organizations therefore want to understand work practice and align it with the organizational objectives. Modeling and simulation can support the description, prescription, and prediction of work practice [3], but also need to show in what ways work practice deviates from the organizational objectives.

Agent-based modeling and simulation used to focus on either the individual, ‘micro’ level in a way that the collective behavior emerges from individual actions (i.e., the bottom-up, *emergent view*), or on the global objectives and desired collective behavior at a ‘macro’ level (i.e., the top-down, *organizational view*). In order to bridge the gap between what the policy makers of an organization want and what the people do, a modeling and simulation methodology is needed that integrates the organizational and emergent views. This will allow policy makers to analyze effects of the micro on the macro level and vice-versa [4].

Related approaches, such as S-Moise+ [5], RNS2 [6], and [7] are similar to our research, in the sense that all aim to develop organizational models to support different levels of coordination and autonomy. However, the difference is that they aim to develop open, heterogeneous multi-agent systems from an engineering perspective, whereas we aim to develop more realistic models of work practice from a human-centered perspective. The second way in which we differ from these approaches is that we provide means to populate the organization with agents specified in some agent language. Thus, we show how the organizational level can be merged with the emergent level.

In this paper we will show how two multi-agent modeling frameworks, OperA [8], a methodology developed to represent and analyze organizational systems, and Brahms [9], a language developed to describe work practice, have been integrated. The combined agent-oriented system engineering *method* (the first step towards a *methodology*) allows the modeling of both the organizational objectives and the emerging (possibly divergent) work practice. By running simulations using the integrated model, it is possible to determine to what degree the workers achieve the organizational objectives. The results of these simulations are used by both policy makers and workers themselves, to understand, test, and improve work practice.

This paper is organized as follows: in section 2 we will introduce the case of Collaborative (Air) Traffic Flow Management. Section 3 describes OperA and Brahms, and section 4 covers their integration into one method. In section 5 the case study is revisited, and section 6 concludes this paper.

2. Case Study: Simulation

Air traffic in the United States of America (USA) has been projected to increase as much as threefold by the year 2025. A simulation of this level of traffic with the current air traffic systems shows a disproportionate and unacceptable increase in average delay per flight. As a result, NASA is researching new technologies and approaches to handle the problems associated with this projected traffic increase. One promising area is Collaborative Traffic Flow Management (CTFM), which seeks to increase the amount of collaboration between the controllers of the airspace (i.e., the

Federal Aviation Administration (FAA)) and the many airlines that use the airspace to find beneficial solutions to traffic flow problems. A *concept of operations* (i.e., a future work process) has been suggested as a specific way to address the problem [10].

At NASA Ames Research Center, this concept of operations is being evaluated through agent-based modeling and simulation, along with other CTFM concepts [11]. We want to know if the new, planned work processes are effective and feasible for the stakeholders, e.g., the FAA, airlines, and passengers. Most of the work processes have only been described on an abstract level (the organizational view), i.e., the desired outcomes. The specific implementation of the work processes that will lead to these outcomes still needs to be defined. This is not straightforward because it requires the coordination of many collaborative activities among highly specialized people in distributed and heterogeneous organizations. Additionally, it is hard to prescribe an exact work process. People may deviate from the objectives (e.g., file the flight plan of each flight) and violate norms, which are constraints on, or specializations of the objectives (e.g., file the flight plan before the flight takes off). Thus, the *work process* is not necessarily the *work practice*. Therefore, investigating varied work practice implementations and their performance on the organizational objectives supports the evaluation of the CTFM concept of operations. This way, we need to model both the organizational view and the emergent view on the work practice.

The organizational view on Airline Operations Centers' (AOC) work has been derived from documented field observations [12], and modeled in OperA. The desired overall behavior of the AOC is external to, and possibly conflicting with, that of the individual workers. Thus, the actual behavior of the AOC will emerge from the combination of the organizational objectives, determined top-down, and the collective behavior that emerges bottom-up. This creates a need to check conformance of the actual behavior to the desired behavior. Using a (Brahms) simulation of the AOC we can now investigate different conceptual, future work practices and see if they meet the organizational objectives (or not). This way, a simulation of the future operations of the AOC can be used to evaluate and improve proposed future work practice.

3. OperA and Brahms

OperA models can not be simulated without another framework for the specification of the agents' behavior. This is because OperA treats agents partly as 'black boxes', i.e., only the desired outcomes of their behavior are specified. However, the way in which these outcomes should be achieved is not specified. Because of that, an OperA model is not executable on its own. Brahms has been used to implement the agents' behavior. This is called an 'instantiation' or 'population' of the OperA model. This decoupling of the abstract description of the organization and the concrete description of the individuals is useful, because it is in accordance with reality, where different groups of people with different work styles may achieve the same objectives in different ways.

3.1. OperA

The OperA model for agent organizations enables the specification of organizational requirements and objectives, and at the same time allows workers to have the freedom to act according to their own capabilities and demands [8]. An OperA model can be seen as a recipe for collective activity; organizations are described in terms of roles, their dependencies and groups, interactions and global norms and communication requirements. Given that OperA assumes organizations as being open systems, it does not include constructs to the specification of the actual agents, treating them as 'black boxes' that commit to a specific (negotiable) interpretation of the organizational roles. OperA meets the following requirements:

- **Internal autonomy requirement**: The internal behavior of the participating agents should be represented independently from the structure of the society.
- **Collaboration autonomy requirement**: The external behavior of the participating agents should be specified without completely fixing the interaction possibilities in advance.

The OperA framework consists of three interrelated models. The **Organizational Model (OM)** is the result of the observation and analysis of the domain and describes the desired behavior of the organization, as determined by the organizational stakeholders in terms of objectives, norms, roles, interactions and ontologies. The **Social Model (SM)** maps organizational roles to specific agents. Agreements concerning the roles an agent will play and the conditions of the participation are described in social contracts. The **Interaction Model (IM)** specifies the interaction agreements between role-enacting agents as interaction contracts.

A generic methodology to determine the type and structure of an application domain is described in [8]. Organizational design starts from the identification of business strategy, stakeholders, their relationships, goals and requirements. It results in a comprehensive organizational model including roles, interactions, objectives, and norms, which fulfill the requirements set by the business strategy. A brief summary of the methodology is given in Table 1.

Table 1. Overview of OperA methodology

	Step	Description	Result
OM	Coordination Level	Identifies organization's main characteristics: purpose, relation forms	Stakeholders, facilitation roles, coordination requirements
	Environment Level	Analysis of expected external behavior of system	Operational roles, use cases, normative requirements
	Behavior Level	Design of internal behavior of system	Role structure, interaction structure, norms, roles, scripts
SM	Population Level	Design of enactment negotiation protocols	Agent admission scripts, role enactment contracts
IM	Interaction Level	Design of interaction negotiation protocols	Scene script protocols, interaction contracts

Organizational Models should take three dimensions of the organization into account: (1) the functional, i.e., plans, objectives, activities, (2) the structural, i.e., roles and groups, (3) the deontic, i.e., norms and constraints [5]. Only if all three dimensions are represented comprehensive organizational models can be built. There is however another requirement for an Organizational Modeling language that can be used for work practice modeling: it needs to allow the specification and participation of autonomous agents, because people are autonomous in most real world situations. OperA fulfills this requirement, because it only specifies the desired outcomes of the activities of the agents and because it takes a deontic perspective in which norms can be violated possibly resulting in sanctions. Similar approaches that can be investigated in the future are S-Moise+ [5] and RNS2 [6]. Other approaches, like ISLANDER [13], do not meet these requirements because either the agent population is specifically generated to fulfill the norms or because there is a control mechanism that blocks all disallowed activities. This is useful for the modeling of electronic institutions, but is not suitable for work practice modeling.

3.2. Brahms

Modeling and simulating work processes is often done at such an abstract level that individual work practice, such as collaboration, communication, ‘off-task’ behaviors, multi-tasking, interrupted and resumed activities, informal interactions, use of tools and movements, is left out, making the description of how the work in an organization actually gets done impossible. The Brahms modeling language is geared towards modeling people’s activity behavior, making it an ideal environment for simulating organizational processes at a level that allows the analysis of the work practice and designing new work processes at the implementation level [3,9].

The Brahms framework consists of several interrelated models. The **Agent Model** describes the behavior of individuals (i.e., people) and groups of individuals (i.e., communities of practice). Individuals are members of groups and inherit the behavior of the groups. Individuals can also have additional behavior that distinguishes them from other individuals, and they can be a member of multiple groups (i.e., multiple inheritance). Groups can be organized in a hierarchical way, to define behavior at different levels of abstraction. Sub-groups inherit the behavior of super-groups. This is convenient for the modeling of common objectives and activities, and does not limit the agents’ autonomy because anything specified on the group level can be overloaded on the agent level. The **Object Model** describes non-cognitive objects (i.e., things). Objects can be physical, or conceptual. The latter means that they only exist within the minds of agents, and can therefore not influence and react on the world. The **Knowledge Model** describes the reasoning of agents and objects, which is based on beliefs and facts. Beliefs are propositions that *represent* the world state and are internal to the agent or object. Facts are *actual* world states, and are global in the simulation world. The **Activity Model** defines the behavior of agents and objects by means of activities and workframes. Brahms has an activity-based subsumption architecture by which an agent’s activities can be decomposed into sub-activities. Activities can be interrupted and resumed, just as humans can multitask by switching between different activities. Workframes control when activities are executed based on the beliefs of the agent, and on facts in the world. The **Communication Model** defines communication activities between agents and objects. When an agent or object communicates, it either sends or receives beliefs from other agents or objects. The **Geography Model** defines a hierarchy of geographical locations representing the space where activities occur. Agents and objects are located in areas and can move from area to area, possibly carrying other agents or objects, by performing a move activity.

There are several requirements for a work practice modeling language. First, it should be a simulation language, i.e., a language that supports the modeling of time. Second, it should support the modeling of activities rather than goals. Third, it should support subsumption, and reactive behavior. Brahms fulfills these requirements because it is a BDI-like activity language. Brahms differs from Jack and Jade in that Brahms is a compiled declarative agent-oriented language. Brahms differs from Jason in that Jason is a goal-based language, while Brahms is an activity-based language. Jason agents are represented using prescribed problems and plans to solve them. Brahms is a behavioral BDI language based on a reactive subsumption architecture, where competing activities are active at once on multiple levels. This allows for seamless activity switching, based on context information the agent is aware of (i.e., has beliefs about). For a description of different multi-agent languages see [14]. For a discussion of how the Brahms language differs from other BDI languages, see [15].

3.3. Rationale for integration of OperA and Brahms

Although OperA was developed almost 10 years later than Brahms, the **philosophy** behind OperA and Brahms is similar. Brahms was developed because work processes were often modeled too abstract, i.e., formal descriptions of work processes differed too much from the actual work practice. Similarly, OperA tries to bridge the gap between the official and the real-world. However, OperA and Brahms have a different **viewpoint** on the solution to this problem. Brahms tries to bridge the gap between the abstracted and the real work practice *bottom-up*, i.e., by observing and describing the individual behavior of people. The modeler can then observe what collective behavior emerges from the interaction of the individual behavior of the people: the emergent view on agent-based modeling. OperA tries to bridge the gap *top-down*, by describing the objectives of an organization. This way it defines what the result of the emergent behavior of the collective should be, rather than describing the practice (i.e., the individual activities and interactions) that should lead to that result: the organizational view on agent-based modeling.

The difference in the viewpoints becomes clear when we compare the **models** that result from the different methodologies. Brahms mainly consists of agents that reason (Knowledge Model) and work (Activity Model). These are definitions of the work that gets done, rather than the results that should be achieved. OperA defines roles, objectives, and norms (Organizational Model). It also defines social contracts (Social Model), which allow the modeler to define which particular agent executes which roles, and which special norms apply. Finally, it defines interaction contracts (Interaction Model), which allow the modeler to describe norms that apply when two or more specific agents, enacting specific roles, interact. These are definitions of (the restrictions on) the results that should be achieved, rather than definitions of the process itself. This shows that OperA and Brahms are orthogonal in this respect.

OperA and Brahms are different **languages**. Brahms is an implementation language. It is formal, and can be compiled to Java, and executed using the Java virtual machine. OperA is a conceptual language, which level of formality depends on the preferences of the modeler and on the development state of the model. Modeling can start by defining objectives and norms in terms of natural language, and then move gradually to pseudo-logic and finally to deontic logic. OperA semantics are formally grounded on the temporal deontic logic LCR [16].

While Brahms is mainly a language, OperA is more of a **methodology** because it provides guidelines on how to get from abstract definitions of work processes (i.e., objectives and norms) to more specific definitions of work processes (i.e., social contracts). This way, there is an order in the models that are created, while in Brahms this is completely up to the modeler.

4. Integration of OperA and Brahms

Based on the complementary viewpoints of OperA and Brahms described in the previous section, we hypothesized that, after integration, the two frameworks could complement each other in the following two ways: (1) OperA adds the top-down (organizational) view to Brahms, Brahms adds the bottom-up (emergent) view to OperA, so that both perspectives are represented, (2) simulations can be run that show the gap between the two perspectives. In order to realize point 2, it is necessary to first convert the OperA model to Brahms, and then to implement the actual work practice. This is done by filling in the specific behavior of the agents, which were treated as ‘black boxes’ in the OperA model. This results in a model that is completely described in Brahms, represents both the organizational view and the emergent view, and which is executable for simulation. In the following, we will discuss how Brahms has been integrated with OperA in a way that meets these requirements.

4.1. Language Integration

OperA consists of three main models: OM, SM and IM. Each of these models is further subdivided into levels and structures (Table 1). If we break these constructs further down we get OperA’s atomic constructs, some of which have been listed in Table 2. In order to be able to convert an OperA model to a Brahms model, we have defined Brahms equivalents for each of the OperA constructs (also in Table 2). Sometimes an OperA construct can be represented by a single, simple Brahms construct, other times several interrelated constructs are needed. Currently, we have defined almost all mappings, without any major difficulties. We believe that the mappings can be done automatically, but future work is necessary to make this possible.

Table 2. Some OperA constructs with Brahms equivalents

OperA	**Brahms**
role `Dispatcher ∈ Roles`$_O$	group 'roles' with sub-groups for each OperA role `group Roles {}` `group Dispatchers memberof Roles {}`
group `{Planners} ⊆ Groups`$_O$	group 'groups' with sub-groups for each OperA group `group Groups {}` `group Planners memberof Groups {}`
objective `safety_ensured(F) ∈ Objectives`$_{Dispatcher}$	workframe on group level + implementation on agent level `group Dispatchers memberof Roles { workframe wf_safety_ensured(Flight flight){ do { pa_safety_ensured(Flight flight) }}}` `agent Deanna memberof Dispatchers { primitive_activity pa_safety_ensured(Flight flight){ // something to ensure safety, and conclude(flight.safety_ensured = true); }}`
norm `IF will_fly(F) THEN OBLIGED released(F)`	workframes + create violation object, on role level `group Dispatchers memberof Roles { workframe wf_norm { when(knownval(flight.will_fly = true) and knownval(flight.released = false)) do { co_norm_violation(Agent agent, Norm norm, Situation situation, Time time); }}}` `// complying agent` `agent Deanna memberof Dispatchers { workframe wf_flight_released { when(knownval(flight.will_fly = true)) do { pa_released(Flight flight); }}}` `// violating agent` `agent Dave memberof Dispatchers { workframe wf_coffee_drunk { when(knownval(flight.will_fly = true)) do { pa_drink_coffee(); }}}`
scene script `Π = file(Dispatcher, P)` `Pattern(Π {DONE(plan_made(Planner, P) BEFORE DONE(plan_sent(Planner, Dispatcher, P) BEFORE plan_filed(Dispatcher, P)}`	workframes + activities, on role level `group Planners memberof Roles { workframe wf_plan_made { when(knownval(plan.made = true)) do { sendToDispatcher(Plan plan); }}}` `group Dispatchers memberof Roles { workframe wf_plan_sent { when(knownval(plan.sent = true)) do { file(Plan plan); }}}`
social contract `SContract(Pete, Planner, {FORBIDDEN(pilot_contacted),…})`	agent membership of role-group, norm on agent level `agent Pete memberof Planners { workframe wf_social_contract { when(knownval(pilot.contacted = true)) do { co_norm_violation(…); }}}`

interaction contract
 	
`IContract(Deanna, Pete, Scene, {…})`
	workframes + activities, on agent level, based on interactions with other specific agents
	
	`agent Deanna memberof Roles {`
	`  workframe wf_interaction_contract {`
	`    when(knownval(plan.sendby = "Pete"))`
	`    do { revise(plan); }}}`

Space and scope limitations do not allow for an in-depth foundation and discussion of the complete mapping. We suffice with an example of one of them, to show that the mapping meets the internal autonomy requirement. Table 2 > row ‘objective’, shows a part of the OperA role definition of the dispatcher, which is a common role in Airline Operation Centers [12]. Dispatchers’ main task is to ensure the safety of flights, which encompasses (among many other things) filing flight plans and releasing flights. The code shows that the objective is defined on the role level, and the activity implementing the work practice is defined on the agent level. This way the agent determines *how* the activity will be executed, and as such, *how* the objective will be met. For even more autonomy, it is possible to specify a workframe with a ‘when’ condition on the agent level. In that case the agent can also determine *when* the activity will be executed and as such *when* the objective will be met. This mapping meets OperA’s internal autonomy requirement, as it allows the designer to define the individual behavior of agents independently from the desired behavior that is defined on the role (organizational) level.

4.2. Guidelines for Methodological Integration

Given the different viewpoints of OperA and Brahms, it can be expected that developers familiar with one or the other framework are used to follow different methodological approaches to system design. In this section, we give some guidelines towards a methodology for the integration of OperA and Brahms models. The guidelines aim to support the modeler in determining which step should be taken in which circumstance. They describe the direction of conversion, the order of development, and the evaluation process for the resulting combined model.

- **Modeling direction**
  - *OperA to Brahms:* Used for simulation of norm-based behavior for different possible populations. This is the direction taken in the case study (Figure 1).
  - *Brahms to OperA:* Used when the goal is to *derive* norms from existing or emerging work practices.
- **Development order**
  - *Breadth strategy:* OperA models are converted to Brahms after each level of modeling. Because Brahms can be executed, it is easier to verify and validate, and thus to find errors. Therefore we recommend converting to Brahms after each model has been developed in OperA.
  - *Depth strategy:* First all OperA models are developed and then converted at once into Brahms. Prevents the modeler from switching her mindset between OperA and Brahms, and is useful for complex mapping processes.

- **Evaluation**
  - *Verification:* **C**heck whether the model has been designed according to the specification. Verification should always precede validation to avoid spread of verification errors.
  - *Validation:* Check whether the model accurately represents reality and thus allows for a realistic simulation. Subject-Matter Experts can support validation. To prevent errors from continuing in later models, validation should happen as frequently as possible.

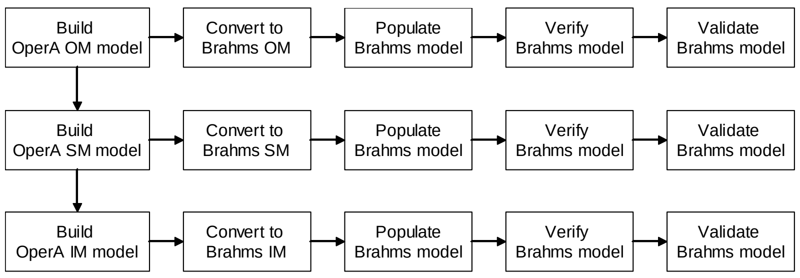

Fig. 1. Describes the modeling direction 'OperA to Brahms' using a breath first strategy, which is the one used in the case study as described in section 5.

5. Case Study: Validation

The integration of OperA and Brahms makes it possible to run a simulation in Brahms in which both the organizational view (e.g., roles, objectives, norms) and the emergent view (e.g., activities, workframes) are represented. But these two views are not necessarily aligned: the work practice may differ from the intentions of the organization's policy makers. Objectives may not be met, and norms may be violated. We have therefore extended the simulation with a monitoring agent, which can detect norm violations. (This is the simplest way, but in the future we would like to support different types of organizations that require agents to monitor themselves or each other.) This makes it possible to perform *norm-based evaluation* of the proposed CTFM operational concepts, which are usually described from the organizational perspective. First an OperA model of these operational concepts was developed. Second this model was converted to Brahms following the guidelines described in section 4. Third the resulting Brahms model was populated with different types of agents, some working according to the objectives and norms defined in the organizational model, and some which prefer their own objectives.

The code in Table 2 > row 'norm' shows the conversion of an OperA norm into Brahms code. The norm holds for all Dispatchers and has thus been defined on the role level. Two agents have been defined, one complying with the norm, and the other violating the norm. When a norm is violated an object is created that contains the identifier of the norm, the situation in which it has been violated (e.g., for which flight), the moment in time, and the agent by which it has been violated. The monitoring agent detects the creation of the norm violation objects, reads out their contents, and reports them to the user of the simulation. The user can then determine the frequency and type of the violations, and which agents are more likely to violate which norms. These simulations show to what degree the actual work practice differs from the organizational objectives. Eventually, this information can be used by policy makers and workers themselves to change organizational objectives and norms, or to improve the work practice.

6. Conclusions

The design and evaluation of work systems can be supported with agent-based modeling and simulation that incorporates both an organizational view (top-down) and an emergent view (bottom-up) on work practice. Most current modeling and simulation frameworks only focus on either one of these views. Therefore we have integrated two frameworks, each representing one of the two views. The integrated method makes it possible to simulate work practice, and to monitor the gap between the emergent behavior and the desired outcomes as defined by the organization's policy makers. This method has been applied to the CTFM concept of operations, by investigating which population and specification of agents' activities and interactions leads to the desired objectives, and conversely, which objectives can be met, based on a certain work practice.

The hypotheses for this research where that the integration of OperA and Brahms meets the following requirements: (1) OperA adds the top-down (organizational) view to Brahms, Brahms adds the bottom-up (emergent) view to OperA, so that both perspectives are represented, (2) simulations can be run that show the gap between the two perspectives. The work presented in this paper provides a proof of concept that supports the validity of these hypotheses. However, more work is needed to improve the integrated method, to provide formal support to these hypotheses, and to determine the actual usefulness of norm-based evaluation of operational concepts.

This work contributes to our more general research objective: How can we improve models of work practice by incorporating the organizational view?, What happens when agents become aware of the fact that they are violating a norm?, What is the influence of norms on work practice?, and: How do norms arise from work practice? New insights in these areas will lead to more realistic models of work practice, and thereby to improved agent-oriented system engineering methodologies.

Acknowledgements
This material is based upon work supported by NASA under award NNA07BB97C. The research of V. Dignum is funded by the Netherlands Organization for Scientific Research (NWO), through Veni-grant 639.021.509.

References

1. Clancey, W.J., Sachs, P., Sierhuis, M., Van Hoof, R.: Brahms: Simulating Practice for Work Systems Design, International Journal on Human-Computer Studies 49, 831--865 (1998)
2. Colvin, A., Boswell, W.: The Problem of Action and Interest Alignment: Beyond Job Requirements and Incentive Compensation. Human Resource Management Review 17, 38--51 (2007)
3. Sierhuis, M.: Modeling and Simulating Work Practice; Brahms: A Multiagent Modeling and Simulation Language for Work System Analysis and Design. PhD Thesis, University of Amsterdam, SIKS Dissertation Series No. 2001-10 (2001)
4. Dignum, V., Dignum, F., Jonker, C.M.: Towards Agents for Policy Making. In: N. David, J. Sichman (Eds.) 9th International Workshop on Multi-Agent-Based Simulation (MABS@AAMAS'08), May 12th, Estoril, Portugal.
5. Hübner, J., Sichman, J., Boissier O.: S-MOISE+: A middleware for developing organised multi-agent systems. In: Boissier, O. et al. (eds.): Coordination, Organizations, Institutions and Norms in MAS (COIN I), volume 3913 of LNCS. Springer (2006)
6. Weiss, G., Nickles, M., Rovatsos M., Fischer, F.: Specifying the intertwining of coordination and autonomy in agent-based systems. International Journal of Network and Computer Applications. Vol.29, (2006)
7. Van der Vecht, B., Dignum, F., Meyer, J.-J.Ch., Neef, M.: A Dynamic Coordination Mechanism Using Adjustable Autonomy. In: COIN@MALLOW'07, Durham, UK, (2007)
8. Dignum, V., Dignum, F., Meyer, J. J.: An Agent-Mediated Approach to the Support of Knowledge Sharing in Organizations. Knowledge Engineering Review 19(2), pp. 147--174 (2004)
9. Sierhuis, M., Clancey, W.J., Van Hoof, R.: Brahms: A Multiagent Modeling Environment for Simulating Work Processes and Practices. International Journal of Simulation and Process Modelling 3(3), 134--152 (2007)
10. Idris, H., Vivona, R., Penny, S., Krozel, J., Bilimoria, K.: Operational Concept for Collaborative Traffic Flow Management based on Field Observations. In: 5th AIAA Aviation, Technology, Integration, and Operations Conference (2005)
11. Wolfe, S.R.: Supporting Air Traffic Flow Management with Agents. In: AAAI Spring Symposium: Interaction Challenges for Intelligent Assistants, Stanford, CA. (2007)
12. Idris, H., Evans, A., Vivona, R., Krozel, J., Bilimoria, K.: Field Observations of Interactions between Traffic Flow Management and Airline Operations. In: 6th AIAA Aviation, Technology, Integration, and Operations Conference, Wichita, Kansas. (2006)
13. Esteva, M., De La Cruz, D., Sierra, C.: ISLANDER: an Electronic Institution Editor. In: 1st Int'l Joint Conf. on Autonomous Agents & Multi-Agent Systems (2002)
14. Bordini, R.H., Dastani, M., Dix, J., Seghrouchni, A.E.F.: Multi-Agent Programming: Languages, Platforms and Applications. New York, NY: Springer Science+Business Media, Inc. (2005)
15. Sierhuis, M.: It's not just goals all the way down – It's activities all the way down. In: O'Hare, G.M.P., Ricci, A., O'Grady, M.J., Dikenelli, O. (eds.) Engineering Societies in the Agents World VII, 7th International, Workshop, ESAW 2006, Dublin, Ireland. (2007)
16. Dignum, V., Meyer, J.J., Dignum, F., Weigand, H.: Formal Specification of Interaction in Agent Societies. In: M. Hinchey et al. (eds.) Formal Approaches to Agent-Based Systems, LNAI 2699, Springer (2003)